

## MODULARITY OF DIRECTED NETWORKS: CYCLE DECOMPOSITION APPROACH

NATAŠA DJURDJEVAC CONRAD

Freie Universität Berlin  
Department of Mathematics and Computer Science  
Arnimallee 6, 14195 Berlin, Germany  
Zuse Institute Berlin  
Takustrasse 7, 14195 Berlin, Germany

RALF BANISCH

Freie Universität Berlin  
Department of Mathematics and Computer Science  
Arnimallee 6, 14195 Berlin, Germany

CHRISTOF SCHÜTTE

Freie Universität Berlin  
Department of Mathematics and Computer Science  
Arnimallee 6, 14195 Berlin, Germany  
Zuse Institute Berlin  
Takustrasse 7, 14195 Berlin, Germany

**ABSTRACT.** The problem of decomposing networks into modules (or clusters) has gained much attention in recent years, as it can account for a coarse-grained description of complex systems, often revealing functional subunits of these systems. A variety of module detection algorithms have been proposed, mostly oriented towards finding hard partitionings of undirected networks. Despite the increasing number of fuzzy clustering methods for directed networks, many of these approaches tend to neglect important directional information. In this paper, we present a novel random walk based approach for finding fuzzy partitions of directed, weighted networks, where edge directions play a crucial role in defining how well nodes in a module are interconnected. We will show that cycle decomposition of a random walk process connects the notion of network modules and information transport in a network, leading to a new, symmetric measure of node communication. Finally, we will use this measure to introduce a communication graph, for which we will show that although being undirected it inherits all necessary information about modular structures from the original network.

**1. Introduction.** In recent years complex networks have become widely recognized as a powerful abstraction of real-world systems [18]. The simple form of network representation allows an easy description of various systems, such as biological, technological, physical and social systems [19]. Thus, network analysis can be used as a tool to understand the structural and functional organization of real-world systems. One of the most important network substructures are **modules** (or

---

2010 *Mathematics Subject Classification.* Primary: 60J20, 05C81; Secondary: 94C15.

*Key words and phrases.* directed networks, modules, cycle decomposition, measure of node communication.

clusters, community structures), that typically correspond to groups of nodes which are similar to each other in some sense. In the case of undirected networks, modules represent densely interconnected subgraphs having relatively few connections to the rest of the network [9]. Finding modules leads to a coarse-grained description of network by smaller subunits, which in many cases have been found to correspond to functional units of real-world systems, such as protein complexes in biological networks [4].

Although a large number of different clustering algorithms [8, 18] have been developed, most of them are oriented towards: (i) finding hard (or full, complete) partitions of networks into modules, where every node is assigned to exactly one module; (ii) analyzing undirected, unweighted networks. However, in many networks a *fuzzy partition*, where some nodes belong to several modules at once or to none at all, might be more natural. Moreover, many real-world networks are weighted and directed [3, 15, 16] such as metabolic networks, email networks, World Wide Web, food webs etc. In such networks edge directions contain important information about the underlying system and as such should not be dismissed [13]. So far, several approaches for finding modules in directed networks have been developed [14, 27, 15, 25], see [16] for an extensive overview. Among these, generalizations of modularity optimization became the most popular group of approaches [1, 15, 12]. We will refer to this family of methods as **generalized Newman-Girvan (gNG)** approaches. Despite their frequent application, modularity based methods are typically only sensitive to the density of links, but tend to neglect information encoded in the pattern of the directions [13, 24]. This limitation of gNG methods (but also one-step methods and methods based on network symmetrization) is illustrated with the following example.

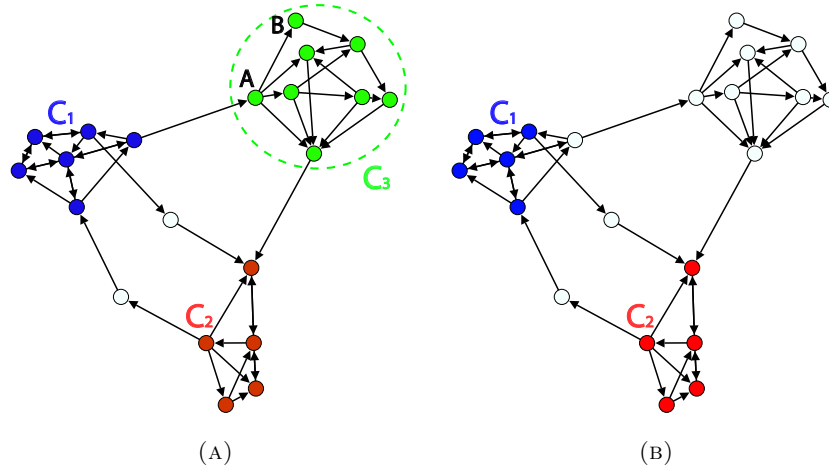


FIGURE 1. Illustration of a false module identification. Colors indicate modules found by (a) gNG algorithm. (b) our new algorithm.

Using a gNG approach, the network shown in Figure 1a is partitioned into three modules  $C_1$ ,  $C_2$  and  $C_3$  with high link density. However,  $C_3$  is qualitatively different from  $C_1$  and  $C_2$ : While in  $C_1$  and  $C_2$  edge directions are fairly symmetric, in  $C_3$  they are completely asymmetric such that all nodes in  $C_3$  are connected to

each other by short paths in one direction and long paths in the other direction. For example, nodes  $A, B \in C_3$  are connected by a directed edge  $(AB)$ , but in the direction from  $B$  to  $A$  they are connected only by long paths that pass through the whole network. In this paper, we will require nodes in a module to be close in both directions and therefore wish to reject a structure such as  $C_3$  from being modular. The result of the algorithm we present here is shown in Figure 1b.

The main topic of this paper is on finding modules in directed, weighted networks, while explicitly respecting information encoded in the pattern of the directions. We will show that the notion of modules in directed networks is closely related to network cycles and described by dynamical properties of a random walk process, see Sections 2.2 and 2.3. We will make this connection precise in Section 2.4, by using a cycle decomposition of this Markov process to introduce a novel measure of communication  $I_{xy}$  between nodes. In Section 2.5, we will demonstrate that  $I_{xy}$  indeed reflects information transport in the network and as such we will use  $I_{xy}$  to introduce a communication graph which, although being undirected, inherits all necessary information about modular structures from the original network by construction. This will allow using some of the known fuzzy clustering approaches on the undirected communication graph, that will produce modules of the original graph. In Section 3 we will describe our new algorithmic approach and its relevant computational aspects. Finally, we will demonstrate our method on several examples in Section 4 and compare it to gNG methods, indicating a possible improvement of modularity function.

## 2. Theoretical background.

**2.1. Random walks on directed networks.** We consider the weighted, directed network  $G = (V; E)$ , where  $V$  is the set of nodes and  $E$  the set of edges. We further assume that the network is strongly connected, i.e. there is a directed path from every node to every other node in the network. In particular, this means that the network has no sinks and sources. Now, we can define a time-discrete random walk process  $(X_n)_n$  on the network with a transition matrix  $P$

$$p_{xy} = \frac{K(x, y)}{K^+(x)}, \quad (1)$$

where  $K(x, y)$  denotes the weight of an edge  $(xy) \in E$  and  $K^+(x) = \sum_y K(x, y)$  the weighted out-degree of a node  $x$ . Since the networks we consider are strongly connected, the introduced random walk process is ergodic and has a unique stationary distribution  $\pi$  with  $\pi^T P = \pi^T$ . However, this process is not time-reversible and the detailed balance condition

$$\pi_x p_{xy} = \pi_y p_{yx}, \quad x, y \in V$$

doesn't hold. For this reason, most of the existing spectral methods fail to deal with finding modules in directed networks, as they are restricted to analysis of reversible processes.

**2.2. Cycle decomposition of flows.** In this section we will briefly introduce the theory of cycle decompositions of Markov processes on finite state spaces as developed in [11] and [10]. We will use this decomposition to describe the random walk process defined by (1) using cycles.

Let us introduce a **probability flow**  $F : E \rightarrow \mathbb{R}$ , such that  $F_{xy} = \pi_x p_{xy}$  for every

edge  $(xy) \in E$ . Then, the equation  $\pi^T P = \pi^T$  of stationarity for  $\pi$  directly yields conservation of the flow at every node  $x$

$$\sum_y F_{yx} = \sum_y F_{xy}. \quad (2)$$

Now we will generalize the notion of edge flows to cycle flows, where we define cycles in the following way

**Definition 2.1.** An  $n$ -cycle (or  $n$ -loop) on  $G$  is defined as an ordered sequence<sup>1</sup> of  $n$  connected nodes  $\gamma = (x_1, x_2, \dots, x_n)$ , whose length we denote by  $|\gamma| = n$ . Cycles that do not have self-intersections are called **simple cycles** and we denote with  $\mathcal{C}$  the collection of simple cycles in  $G$ .

In Figure 2 are shown examples of two simple cycles  $\alpha$  and  $\beta$ , where  $|\alpha| = 4$  and  $|\beta| = 2$ .

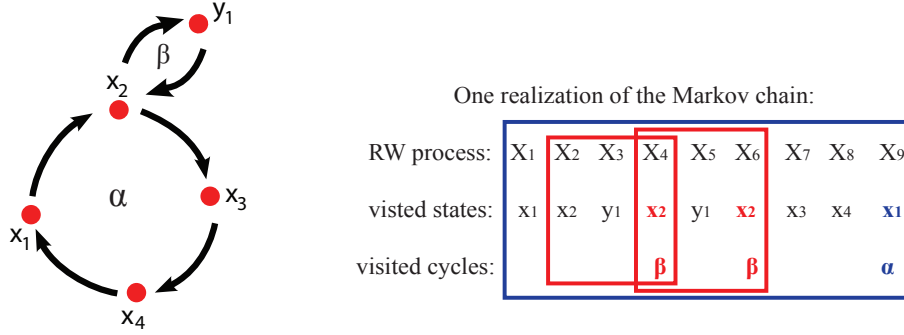


FIGURE 2. Examples of two simple cycles  $\alpha$  and  $\beta$  and one realization of the Markov chain  $(X_n)_x$ .

As a consequence of ergodicity and the flow conservation law (2), we can find a cycle decomposition of  $F$  into cycles. More precisely, there is a collection  $\Gamma \subset \mathcal{C}$  of cycles  $\gamma$  with real positive weights  $F(\gamma)$  such that for every edge  $(xy)$  the flow decomposition formula holds

$$F_{xy} = \sum_{\gamma \supset (xy)} F(\gamma), \quad (3)$$

where we write  $\gamma \supset (xy)$  if the edge  $(xy)$  is in  $\gamma$ . Different approaches for finding cycle decompositions of Markov processes [26, 28, 11, 2] have been developed. In this paper we will refer to a **stochastic decomposition approach** introduced in [11, 10], as this approach leads to finding a unique cycle decomposition. Let  $(X_n)_{1 \leq n \leq T}$  be a realization of the Markov chain given by (1). We will say that  $(X_n)_{1 \leq n \leq T}$  **passes through the edge**  $(xy)$  if there exists  $k < T$  such that  $X_k = x$  and  $X_{k+1} = y$ . The process  $(X_n)_{1 \leq n \leq T}$  **passes through a cycle**  $\gamma$  if it passes through all edges of a cycle  $\gamma$  in the right order, but not necessarily consecutively meaning that excursions through one or more full new cycles are allowed. This

<sup>1</sup>More precisely, cycles are equivalence classes of ordered sequences up to cycle permutations. In this note we do not distinguish between cycles and their representatives.

behavior is illustrated in Figure 2.

Let  $N_T^\gamma$  be the number of times  $(X_n)_{1 \leq n \leq T}$  passes through a cycle  $\gamma$  up to time  $T$ . The limit

$$w(\gamma) := \lim_{T \rightarrow \infty} \frac{N_T^\gamma}{T} \quad (4)$$

exists almost surely [10]. Then, if the following **weights condition (WC)** holds

$$\mathbf{WC} : \Gamma = \{\gamma | w(\gamma) > 0\}, \quad F(\gamma) = w(\gamma), \quad (5)$$

we obtain a unique cycle decomposition  $(\Gamma, w)$ . In terms of the random walk process, the weight  $w(\gamma)$  denotes the mean number of passages through a cycle  $\gamma$  of almost all sample paths  $(X_n)_{1 \leq n \leq T}$ . We will refer to  $w(\gamma)$  as an importance weight for the cycle  $\gamma$ .

An explicit formula to calculate the weights  $w(\gamma)$  was given in [10], which reads  $\gamma = (x_1, \dots, x_n)$

$$w(\gamma) = \pi_{x_1} p_{x_1 x_2} \dots p_{x_n x_1} N_{x_1 \dots x_n} \quad (6)$$

where the normalization factor  $N_{x_1 \dots x_n}$  is a product of taboo Green's functions accounting for the excursions the random walk process can take while completing  $\gamma$ . It is hard to calculate  $w(\gamma)$  in general, so we will use (6) for illustration purposes only, see [10] for more details.

As an example we will consider the barbell graph presented in Figure 3. This

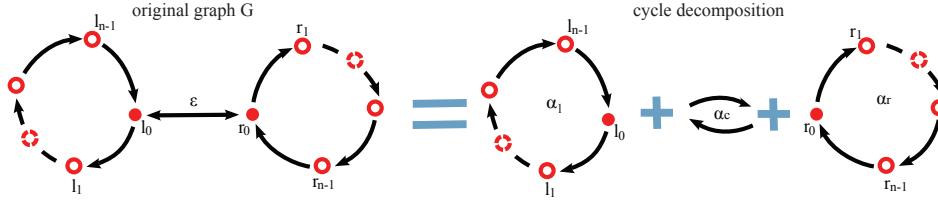


FIGURE 3. The barbell graph example network, consisting of two cycles with  $n$  nodes joined by an edge with weight  $\epsilon$ . The cycle decomposition of the barbell graph consists of three cycles  $\alpha_l$ ,  $\alpha_c$  and  $\alpha_r$ .

is a weighted, directed graph consisting of two cycles  $\alpha_l = (l_0, l_1, \dots, l_{n-1})$  and  $\alpha_r = (r_0, r_1, \dots, r_{n-1})$  with  $n$  nodes each and all edge weights are one. These two cycles are joined by an undirected edge  $\alpha_c = (l_0, r_0)$  with edge weight  $K(l_0, r_0) = K(r_0, l_0) = \epsilon < 1$ . Then, the probability of a standard random walk process (1) leaving any of the two cycles at  $l_0$  or  $r_0$  is  $p_{l_0 r_0} = \frac{\epsilon}{1+\epsilon} = p_{r_0 l_0}$ . The central nodes have the same stationary distribution  $\pi_{l_0} = \pi_{r_0} = \pi_c$  and for all other nodes we have  $\pi_{l_i} = \pi_{r_i} = \pi_l$ . Using the flow conservation property (2) at one of the central nodes we get the equation  $\pi_l + \pi_c \frac{\epsilon}{1+\epsilon} = \pi_c$  from which we can easily calculate

$$\pi_l = \frac{1}{2(n+\epsilon)}, \quad \pi_c = (1+\epsilon)\pi_l = \frac{1+\epsilon}{2(n+\epsilon)}.$$

Since every edge belongs to exactly one of the three loops  $\alpha_c = (l_0, r_0)$ ,  $\alpha_l = (l_0, l_1, \dots, l_{n-1})$  and  $\alpha_r = (r_0, r_1, \dots, r_{n-1})$ , the weights of these loops can be inferred directly from (3) as

$$w(\alpha_l) = w(\alpha_r) = \frac{1}{2(n+\epsilon)} =: w, \quad w(\alpha_c) = \epsilon w.$$

**2.3. Cycle-based random walks.** In this section, we will use the stochastic cycle decomposition to introduce reversible processes that are a good approximation of the original non-reversible process, in the sense that they contain all the important directional information needed for the module identification. This will allow us to construct graph transformations which will map a directed graph into an undirected graph, such that cycle node connections (i.e. if  $x, y \in \alpha$ ) and the cycle importance weights are encoded into the structure of the new graph. Then, we will be able to reduce the original module finding problem on directed networks to a well understood problem of finding module partitions of undirected networks.

We start by reformulating the original dynamics  $(X_n)_n$  on vertices  $V$  in terms of a dynamics on the cycle space  $\Gamma$ . Namely, instead of the dynamics of the standard random walk process with jumps from one node to another node, we will look at the random walk process with dynamics of the type node-cycle-node. To this end, we introduce

**Definition 2.2.** The process  $(\tilde{X}_n)_n$  is a stochastic process on  $V$  generated by the following procedure:

1. If  $\tilde{X}_n = x$ , then draw the next cycle  $\xi_n$  according to the conditional probabilities

$$\mathbf{P}(\xi_n = \alpha | \tilde{X}_n = x) =: b_\alpha^{(x)} = \begin{cases} \frac{1}{\pi_x} w(\alpha), & x \in \alpha \\ 0 & x \notin \alpha. \end{cases}$$

2. If  $\xi_n = \alpha$ , then follow the cycle  $\alpha$  one step to obtain  $\tilde{X}_{n+1} = y, (xy) \subset \alpha$ .

In this process, jumps from a node  $x$  to a cycle  $\alpha$  are probabilistic with probabilities  $b_\alpha^{(x)}$  given by the cycle decomposition  $\{\Gamma, w\}$ . Using (3) it follows that  $\sum_{\alpha \supset x} w(\alpha) = \pi_x$ , so that  $b_\alpha^{(x)}$  form indeed conditional probabilities. Furthermore,  $(\tilde{X}_n)_n$  is a Markov process with a transition matrix  $P$

$$\begin{aligned} \mathbf{P}(\tilde{X}_{n+1} = y | \tilde{X}_n = x) &= \sum_{\alpha \supset (xy)} \mathbf{P}(\xi_n = \alpha | \tilde{X}_n = x) \\ &= \frac{1}{\pi_x} \sum_{\alpha \supset (xy)} w(\alpha) \\ &\stackrel{(3)}{=} \frac{1}{\pi_x} F_{xy} = p_{xy}. \end{aligned}$$

Thus,  $(X_n)_n$  and  $(\tilde{X}_n)_n$  are statistically identical and we will refer to them as  $(X_n)_n$  in the rest of the paper. As a byproduct we obtained the associated chain  $(\xi_n)_n \subset \Gamma$  of cycles used by  $(X_n)_n$ . Notice that cycle-node jumps are deterministic and that  $(\xi_n)_n$  is a non-Markovian process. To see this, consider the barbell graph in Figure 3. We have  $\mathbf{P}(\xi_{n+2} = \alpha_r | \xi_{n+1} = \alpha_r, \xi_n = \alpha_c) = 1$  since in this case  $X_{n+2} = r_1$  with probability 1. But  $\mathbf{P}(\xi_{n+2} = \alpha_r | \xi_{n+1} = \alpha_r) = 1 - \frac{1}{n} \frac{\varepsilon}{1+\varepsilon}$  as in this case  $X_{n+2} = r_0$  with probability  $1/n$ , so that  $X_{n+2} = \alpha_c$  is possible.

If  $(X_n)_n$  is in equilibrium such that  $\mathbf{P}(X_n = x) = \pi_x$ , then the chain  $(\xi_n)_n$  is also in equilibrium, with distribution

$$\begin{aligned} \mu(\alpha) &:= \mathbf{P}(\xi_n = \alpha) = \sum_x \mathbf{P}(\xi_n = \alpha | X_n = x) \mathbf{P}(X_n = x) = \sum_{x \subset \alpha} b_\alpha^{(x)} \pi_x \\ &= \sum_{x \subset \alpha} \frac{1}{\pi_x} w(\alpha) \pi_x = |\alpha| w(\alpha). \end{aligned}$$

Further assuming that  $(X_n)_n$  is in equilibrium, the one-step transition matrix of  $(\xi_n)_n$  is

$$\mathcal{Q}_{\alpha\beta} := \mathbf{P}(\xi_n = \beta | \xi_{n-1} = \alpha) = \sum_{x \in (\alpha \cap \beta)} \frac{1}{|\alpha|} b_\beta^{(x)}, \quad (7)$$

see Appendix for detailed calculation. Moreover,  $\mathcal{Q}$  is detailed-balanced

$$\mu(\alpha) \mathcal{Q}_{\alpha\beta} = \sum_{x \in (\alpha \cap \beta)} \frac{1}{\pi_x} w(\alpha) w(\beta) = \mu(\beta) \mathcal{Q}_{\beta\alpha}. \quad (8)$$

Thus, given a non-reversible Markov chain  $(X_n)_n$  on  $V$ , we have constructed a reversible, but non-Markovian associated chain  $(\xi_n)_n$  on the cycle space  $\Gamma$ . We will now modify Definition 2.2 to obtain two Markov chains  $(Z_n)_n \subset V$  and  $(\zeta_n)_n \subset \Gamma$ . It will turn out later that  $(\zeta_n)_n$  is the Markov approximation of  $(\xi_n)_n$ .

**Definition 2.3.** The *cycle-node-cycle* process  $(\zeta_n)_n \subset \Gamma$  and the *node-cycle-node* process  $(Z_n)_n \subset V$  are stochastic processes generated by the following procedure:

1. If  $Z_n = x$ , then draw the next cycle  $\zeta_n$  according to the conditional probabilities

$$\mathbf{P}(\zeta_n = \alpha | Z_n = x) = b_\alpha^{(x)}.$$

2. If  $\zeta_n = \alpha$ , then draw  $Z_{n+1} = y$  with probability

$$\mathbf{P}(Z_{n+1} = y | \zeta_n = \alpha) := \begin{cases} \frac{1}{|\alpha|}, & y \in \alpha \\ 0, & y \notin \alpha. \end{cases}$$

Jumps node-cycle are again probabilistic, but now also jumps from a cycle to a node are probabilistic because from a cycle  $\alpha$  the process can jump with uniform probability to any of the nodes in this cycle. Both processes  $(Z_n)_n$  and  $(\zeta_n)_n$  are Markov chains and we will now look at their transition properties. To simplify the notation we introduce the two stochastic matrices  $\mathcal{B} : \mathbb{R}^\Gamma \rightarrow \mathbb{R}^V$  and  $\mathcal{V} : \mathbb{R}^V \rightarrow \mathbb{R}^\Gamma$  with components

$$\begin{aligned} \mathcal{B}_{x\alpha} &= \mathbf{P}(\zeta_n = \alpha | Z_n = x) = b_\alpha^{(x)}, \\ \mathcal{V}_{\alpha y} &= \mathbf{P}(Z_{n+1} = y | \zeta_n = \alpha) = \frac{1}{|\alpha|} \delta(y \in \alpha). \end{aligned} \quad (9)$$

Let  $\pi_n(x) := \mathbf{P}(Z_n = x)$  and  $\mu_n(\alpha) := \mathbf{P}(\zeta_n = \alpha)$  be the probability distributions of  $(Z_n)_n$  and  $(\zeta_n)_n$ . Then the above definition implies the following propagation scheme for  $\pi_n$  and  $\mu_n$

$$\pi_n \xrightarrow{\mathcal{B}^T} \mu_n \xrightarrow{\mathcal{V}^T} \pi_{n+1} \xrightarrow{\mathcal{B}^T} \mu_{n+1} \xrightarrow{\mathcal{V}^T} \dots \quad (10)$$

The next lemma gives the explicit expressions for the transition matrices of  $(\zeta_n)_n$  and  $(Z_n)_n$  in terms of  $\mathcal{B}$  and  $\mathcal{V}$

**Lemma 2.4.** Let  $\mathcal{P}$  be the transition matrix of  $(Z_n)_n$  and  $\mathcal{Q}$  be the transition matrix of  $(\zeta_n)_n$ . Then  $\mathcal{P} = \mathcal{B}\mathcal{V}$  and  $\mathcal{Q} = \mathcal{V}\mathcal{B}$  and

$$\mathcal{P}_{xy} = \frac{1}{\pi_x} \sum_{\alpha \ni x, y} \frac{w(\alpha)}{|\alpha|} \quad (11)$$

$$\mathcal{Q}_{\alpha\beta} = \sum_{x \in (\alpha \cap \beta)} \frac{1}{|\alpha|} b_\beta^{(x)}. \quad (12)$$

*Proof.* By the Kolmogorov forward equation  $\pi_{n+1} = \mathcal{P}^T \pi_n$ , which in view of (10) immediately implies  $\mathcal{P} = \mathcal{B}\mathcal{V}$ , and similarly we get  $\mathcal{Q} = \mathcal{V}\mathcal{B}$ . On the other hand, using (9):

$$(\mathcal{V}\mathcal{B})_{\alpha\beta} = \sum_{x \in \alpha \cap \beta} b_{\beta}^{(x)} \frac{1}{|\alpha|}, \quad (\mathcal{B}\mathcal{V})_{xy} = \frac{1}{\pi_x} \sum_{\alpha \ni x, y} \frac{w(\alpha)}{|\alpha|}.$$

Since  $\mathcal{B}$  and  $\mathcal{V}$  are stochastic, then  $\mathcal{P}$  and  $\mathcal{Q}$  are stochastic matrices too.  $\square$

Process	$(\mathbf{X}_n)_n$	$(\xi_n)_n$	$(\mathbf{Z}_n)_n$	$(\zeta_n)_n$
Properties	non-reversible Markov	reversible non-Markov	reversible Markov	reversible Markov
State space	$V$	$\Gamma$	$V$	$\Gamma$
Transition matrix	$P$	$\mathcal{Q}$	$\mathcal{P}$	$\mathcal{Q}$
Stationary distribution	$\pi$	$\mu$	$\pi$	$\mu$

TABLE 1. Properties of the introduced random walk processes.

By comparing with (7), we can see that the one-step transition matrices of  $(\zeta_n)_n$  and  $(\xi_n)_n$  are the same, so that  $(\zeta_n)_n$  is the Markov approximation of  $(\xi_n)_n$ . Thus, our construction produced a Markov process  $(Z_n)_n$  on the space of graph nodes  $V$  with transition matrix  $\mathcal{P}$  and a closely related Markov process  $(\zeta_n)_n$  on the space of graph cycles  $\Gamma$  with the transition matrix  $\mathcal{Q}$ . Table 1 shows the main properties of the introduced random walk processes. To establish a more precise relation between these processes, we will examine the spectral properties of  $\mathcal{P}$  and  $\mathcal{Q}$ .

Let  $D_{\pi} = \text{diag}(\pi)$  and  $D_{\mu} = \text{diag}(\mu)$  with  $\mu(\alpha) = |\alpha|w(\alpha)$  as before. We equip  $\mathbb{R}^V$  and  $\mathbb{R}^{\Gamma}$  with the scalar products  $\langle u, v \rangle_{\pi} := u^T D_{\pi} v$  and  $\langle u, v \rangle_{\mu} := u^T D_{\mu} v$ . Then, Lemma 2.5 shows that  $(Z_n)_n$  and  $(\zeta_n)_n$  are two time-reversible Markov processes with identical spectral properties.

**Lemma 2.5.** *The transition matrices  $\mathcal{P}$  and  $\mathcal{Q}$  of processes  $(Z_n)_n$  and  $(\zeta_n)_n$  have the following properties:*

- (i)  $\pi$  is the stationary distribution of  $\mathcal{P}$  and  $\mu$  is the stationary distribution of  $\mathcal{Q}$ .
- (ii)  $(Z_n)_n$  and  $(\zeta_n)_n$  are time-reversible processes, i.e.  $\langle u, \mathcal{P}v \rangle_{\pi} = \langle \mathcal{P}u, v \rangle_{\pi}$  and  $\langle u, \mathcal{Q}v \rangle_{\mu} = \langle \mathcal{Q}u, v \rangle_{\mu}$ .
- (iii) The non-zero spectrum of  $\mathcal{Q}$  is identical to the non-zero spectrum of  $\mathcal{P}$ , i.e.  $\sigma(\mathcal{Q}) \setminus \{0\} = \sigma(\mathcal{P}) \setminus \{0\}$ .
- (iv) For  $\lambda \in \sigma(\mathcal{P}) \setminus \{0\}$ ,  $\dim \ker(\mathcal{P} - \lambda 1) = \dim \ker(\mathcal{Q} - \lambda 1)$ . If  $v \in \ker(\mathcal{Q} - \lambda 1)$ , then  $\mathcal{B}v \in \ker(\mathcal{P} - \lambda 1)$ . If  $v \in \ker(\mathcal{P} - \lambda 1)$ , then  $\mathcal{V}v \in \ker(\mathcal{Q} - \lambda 1)$ .

*Proof.* See Appendix.  $\square$

Figure 4 shows the spectrum of transition matrices  $P$  and  $\mathcal{P}$  for the barbell graph example from Figure 3 and  $n = 40$ ,  $\epsilon = 0.1$ . The spectrum of the transition matrix  $P$  of the original non-reversible random walk process consists of complex eigenvalues which all lie almost on the unit circle, i.e. have a module almost 1. On the other hand, the spectrum of  $\mathcal{P}$  (therefore also  $\mathcal{Q}$ ) is real valued and offers a clear spectral gap after the second eigenvalue, indicating the existence of two network modules. The benefit of the new random walk process in this example is that it preserves the dominant real valued eigenvalues related to the existence of network modules and projects out the complex eigenvalues.



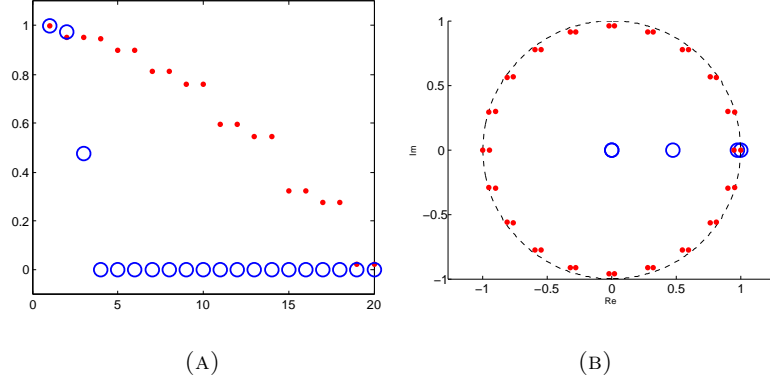


FIGURE 4. The eigenvalues of random walk processes on a barbell graph with  $n = 40$  and  $\epsilon = 0.1$ . Red: Eigenvalues of  $P$ . Blue: Eigenvalues of  $\mathcal{P}$ . 4a shows the real part of the largest 20 eigenvalues of  $P$  and  $\mathcal{P}$ . 4b shows the complete spectrum of  $P$  and  $\mathcal{P}$ .

**2.4. Communication graph and cycle graph.** Time-reversibility of  $(Z_n)_n$  and  $(\zeta_n)_n$  directly implies that the weights  $I_{xy} := \pi_x \mathcal{P}_{xy}$  and  $W_{\alpha\beta} := \mu(\alpha) \mathcal{Q}_{\alpha\beta}$  are symmetric. Remarkably, both  $W_{\alpha\beta}$  and  $I_{xy}$  have an interpretation in terms of the original Markov process  $(X_n)_n$ . More precisely, the quantity

$$W_{\alpha\beta} = \sum_{x \in (\alpha \cap \beta)} w(\alpha) b_\beta^{(x)}, \quad \alpha, \beta \in \Gamma \quad (13)$$

is the total exchange of probability flux  $F$  between the cycles  $\alpha$  and  $\beta$  per timestep. To see this, note that for  $x \in \alpha$ ,  $w(\alpha)$  is the fraction of probability current arriving at  $x$  which is carried by  $\alpha$ , and  $b_\beta^{(x)}$  is the probability to switch to  $\beta$  at  $x$ , so that  $w(\alpha) b_\beta^{(x)}$  is the exchange of probability flux between  $\alpha$  and  $\beta$  that happens at  $x$ . The quantity

$$I_{xy} = \sum_{\alpha \ni x, y} \frac{w(\alpha)}{|\alpha|}$$

is a measure for intensity of communication between nodes  $x, y$  under  $(X_n)_n$  in terms of the cycles through which  $x$  and  $y$  are connected. Indeed,  $I_{xy}$  is large, when  $x$  and  $y$  are connected by many cycles  $\alpha$  which are important (i.e.  $w(\alpha)$  large) and short (i.e.  $|\alpha|$  small). This interpretation of  $I_{xy}$  as a communication intensity measure will be made clear in the next Section 2.5.

We will use these properties of  $W_{\alpha\beta}$  and  $I_{xy}$ , to introduce the following undirected graphs:

**Definition 2.6.** Given the directed graph  $G = (V, E)$ , we let  $H_G(\Gamma, E_W)$  be the undirected graph with vertex set  $\Gamma$  which has an edge with weight  $W_{\alpha\beta}$  between  $\alpha$  and  $\beta$  if  $W_{\alpha\beta} > 0$ , and we call  $H_G$  the **cycle graph** of  $G$ . We let  $K_G(V, E_I)$  be the undirected graph with vertex set  $V$  which has an edge with weight  $I_{xy}$  between  $x$  and  $y$  if  $I_{xy} > 0$ , and we call  $K_G$  the **communication graph** of  $G$ .

Notice that  $H_G$  is simply the network representation of the cycle-node-cycle process  $(\zeta_n)_n$  and  $K_G$  is the network representation of the node-cycle-node process

$(Z_n)_n$ . Thus, the standard random walk processes on  $H_G$  and  $K_G$  will have transition matrices  $\mathcal{Q}$  and  $\mathcal{P}$  respectively. Since important module information from  $G$  is encoded in the structure of both  $H_G$  and  $K_G$ , we can apply some of the existing module finding approaches in undirected graphs on  $H_G$  and  $K_G$  and project the result to  $G$ . In this paper we will present the algorithm for module identification which uses only the communication graph (see Section 3), because  $H_G$  is usually much larger than  $K_G$  for modular directed networks  $G$ .

In Figure 5, the barbell graph is shown together with the corresponding cycle

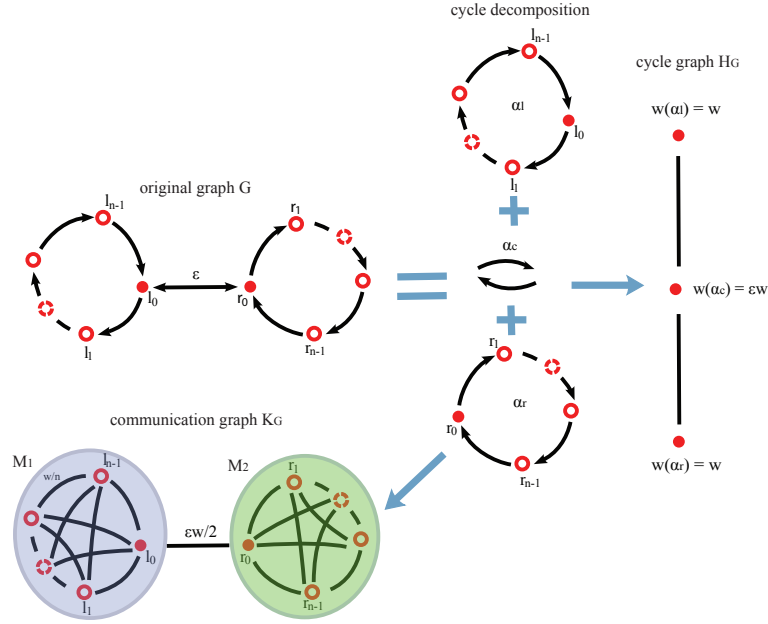


FIGURE 5. The barbell graph and corresponding cycle and communication graphs.

graph  $H_G$  and communication graph  $K_G$ . Since the cycle decomposition of the barbell graph produces three cycles  $\alpha_l$ ,  $\alpha_c$  and  $\alpha_r$ , the cycle graph will have only three nodes. The edge weights (13) of the cycle graph are  $W_{\alpha_l \alpha_c} = \frac{w\epsilon}{1+\epsilon} = W_{\alpha_c \alpha_r}$ . The communication graph of the barbell graph has the following edge weights

$$I_{xy} = \begin{cases} w/n & x, y \in \alpha_l \\ w/n & x, y \in \alpha_r \\ \epsilon w/2 & x \neq y \in \alpha_c \\ \epsilon w/2 + w/n & x = y \in \alpha_c \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

As observed in Figure 5, this results in cycles  $\alpha_l$  and  $\alpha_r$  being mapped into two complete subgraphs with equally weighted edges and consequently the appearance of two modules  $M_1$  and  $M_2$  in  $K_G$ .

**Remark 1.** (Line-graphs) Cycle graphs can be related to line-graphs (or edge-graphs), which are important objects in spectral graph theory [21, 22]. If  $G = (V, E)$  is an undirected graph with  $|E|$  edges, its line-graph  $L(G)$  has  $|E|$  nodes which are edges of  $G$  and nodes in  $L(G)$  are adjacent if edges in  $G$  are. So, a transformation of  $G$  into  $L(G)$  maps graph's edges into nodes and a transformation of  $G$  into  $H_G$  maps graph's cycles into nodes. Furthermore, line-graphs can be very efficiently used for fuzzy clustering of undirected networks [7] by employing some of the full partitioning approaches for finding modules in line-graphs and then projecting these modules back to the original graph. Similarly, one can use cycle graphs for fuzzy partitioning of directed networks.

**Remark 2.** (Entropy production) The work of [2] introduces cycle graphs as a way to describe non-equilibrium Markov processes. Indeed, many non-equilibrium properties may very well be related to cycle decompositions of the type we discussed here. We give one example. The *entropy production rate* is believed to be a measure of the degree of irreversibility of a system. For Markov chains, it is given by

$$e_p = \sum_{i,j} \frac{1}{2} (\pi_i p_{ij} - \pi_j p_{ji}) \log \left( \frac{\pi_i p_{ij}}{\pi_j p_{ji}} \right).$$

It has been shown in [10] that the entropy production rate can be expressed by means of the stochastic cycle decomposition (4) as

$$e_p = \sum_{\gamma \in \Gamma} (w(\gamma) - w(\gamma^-)) \log \left( \frac{w(\gamma)}{w(\gamma^-)} \right).$$

Here,  $\gamma^-$  denotes the cycle  $\gamma$  with reversed orientation. This shows that the entropy production rate is given in terms of the difference of the cycle weights for cycles  $\gamma$  and their reversed counterparts  $\gamma^-$ . More precise relations between cycle decomposition and entropy production will be the topic of our research in the future.

**2.5. Information transport.** In this section we show that the cycle-node-cycle process  $(Z_n)_n$  in fact mimics a model for information transport in the network, which gives an interpretation to the quantity  $I_{xy}$ . Suppose that initially,  $X_0 = x$ . Let  $\tilde{Z}_0 = x$  denote the initial location of one bit of information. We will now model how this bit is transported through the network by means of  $(X_n)_n$ :

1. Let  $\tau$  be the first time  $(X_n)_n$  returns to  $x$ . Since we assumed  $(X_n)_n$  to be irreducible,  $\tau$  is a.s. finite and the path  $p = (X_0, \dots, X_\tau)$  is a loop, possibly with self-intersections. We can decompose  $p$  into simple cycles, exactly one of which contains  $x$ . Let this cycle be  $\gamma$ .
2. The bit of information is now distributed uniformly among all nodes on  $\gamma$ . To model this, we will pick the new location  $\tilde{Z}_1 = y$  of the bit uniformly at random among the vertices of  $\gamma$ . We will then restart the process  $(X_n)_n$  at  $y$  and repeat.

Now observe that the outcome of step 1 is the cycle  $\gamma$  exactly iff  $(X_n)_n$  **passes through**  $\gamma$ , as it was defined in Section 2.2. The probability of the outcome being  $\gamma$  is therefore the probability of  $(X_n)_n$  passing through  $\gamma$  conditioned on starting at  $x$ , which is  $\delta(x \in \gamma)w(\gamma) / \sum_{\gamma \ni x} w(\gamma) = \mathcal{B}_{x\gamma}$ . On the other hand, if the outcome of (1) is  $\gamma$ , then the probability of  $\tilde{Y}_1 = y$  in (2) is  $\frac{1}{|\gamma|} \delta(y \in \gamma) = \mathcal{V}_{\gamma y}$ . We thus have

by Lemma 2.5:

$$\mathbf{P}(\tilde{Z}_1 = y | \tilde{Z}_0 = x) = \sum_{\gamma} \mathcal{B}_{x\gamma} \mathcal{V}_{\gamma y} = \mathcal{P}_{xy} = \mathbf{P}(Z_1 = y | Z_0 = x).$$

This implies that  $(\tilde{Z}_n)_n = (Z_n)_n$ , that is the node-cycle-node process describes the sequence of locations of the bit of information which is transported along the network by  $(X_n)_n$ . Then

$$I_{xy} = \mathbf{P}(Z_n = x, Z_{n+1} = y) = \sum_{\alpha \ni x, y} \frac{w(\alpha)}{|\alpha|}$$

is the probability that the bit is delivered from  $x$  to  $y$ , and therefore measures how easy it is to share information between the nodes  $x$  and  $y$ . From the point of view of the network  $G$ ,  $I_{xy}$  connects topological properties of  $G$  (how many cycles connect  $x$  and  $y$ , and how long are they) with dynamical properties of the process  $(X_n)_n$  (which gives the importance weights  $w(\alpha)$ ). As we established in Section 2.4,  $I_{xy}$  is symmetric since  $(Z_n)_n$  is reversible. This allows us to apply any clustering method designed for undirected, weighted networks to find a partition of  $G$  into modules based on node communication. The algorithmic aspect of this approach will be presented in the next section together with a view on its computational complexity.

### 3. Algorithmic aspects.

**3.1. Identification of modules.** Following the ideas developed in Section 3, we will say that a subset  $M \subset V$  is a module of the directed graph  $G = (V, E)$  iff it is a module in the usual sense of the undirected, weighted *communication graph*  $K_G = (V, E_I)$  defined in Section 2.4. This suggests the following procedure to identify modules in  $G$ :

1. Given the graph  $G = (V, E)$  with edge weights  $K(x, y)$ , construct the transition matrix  $P$  according to (1).
2. Obtain the cycle decomposition  $\{\Gamma, w\}$  of the RW given by  $P$ , where the weights  $w(\gamma)$  satisfy (4).
3. Construct the undirected weighted graph  $K_G$  and find modules in  $K_G$ .

To partition  $K_G$  in step (3), we will use Markov State Modeling (MSM) clustering [5, 23], but any other method designed for clustering undirected, weighted graphs could be used instead [8, 18]. MSM clustering returns  $m$  module cores  $M_1, \dots, M_m \subset V$ , which are disjoint and do not necessarily form a full partition

$$\bigcup_{i=1}^m M_i \neq V \quad \Rightarrow \quad T = V \setminus \bigcup_{i=1}^m M_i \neq \emptyset.$$

The number of modules  $m$  can be inferred from the spectrum of  $P$ , see [6] for more details. Furthermore, the MSM approach aims at finding **fuzzy network partitions** into modules, such that some nodes do not belong to only one of the modules but to several or to none at all. For nodes that do not deterministically belong to one of the modules, we will say that they belong to the **transition region**  $T$ . The next obvious question is how to characterize the affiliation of the nodes from the transition region to one of the modules. In the MSM approach, this affiliation is given in terms of the random walk process  $(Z_n)_n$  on  $K_G$  by the **committor functions**

$$q_i(x) = \mathbf{P} \left[ \tau_x \left( \bigcup_{j \neq i}^m M_j \right) > \tau_x(M_i) \right], \quad i = 1, \dots, m, x \in V$$

where  $\tau_x(A)$  is the first hitting time of the set  $A \subset V$  by the process  $(Z_n)_n$  if started in node  $x$ ,  $\tau_x(A) = \inf\{t \geq 0 : Z_t \in A, Z_0 = x\}$ . Therefore,  $q_i(x)$  gives us the probability that the random walk process if started in node  $x$ , will enter the module  $M_i$  earlier than any other module. Committed functions can be computed very efficiently by solving sparse, symmetric and positive definite linear systems [17, 6]

$$\begin{aligned} (\text{Id} - P)q_i(x) &= 0, \quad \forall x \in T \\ q_i(x) &= 1, \quad \forall x \in M_i \\ q_i(x) &= 0, \quad \forall x \in M_j, j \neq i. \end{aligned} \tag{15}$$

Furthermore, it is easy to see that the committed functions  $q_1, \dots, q_m$  form a partition of unity

$$\sum_{i=1}^m q_i(x) = 1, \quad \forall x \in V,$$

such that we can interpret  $q_i(x)$  as the natural random walk based probability of affiliation of a node  $x$  to a module  $M_i$ .

**3.2. Obtaining the cycle decomposition.** We now turn our attention to the problem of actually obtaining the cycle decomposition  $\{\Gamma, w\}$  with weights  $w(\gamma)$  satisfying (4). This is clearly a hard problem: While even finding and enumerating all cycles in  $G$  scales exponentially in the worst case, explicitly computing the weight  $w(\gamma)$  for  $\gamma = (x_1, \dots, x_l)$  according to (6), requires the computation of  $l$  taboo Green functions of the form  $(I - P_{\{x_1, \dots, x_s\}})^{-1}$  for  $1 \leq s \leq l$  where  $P_{\{x_1, \dots, x_s\}}$  is  $P$  with rows and columns indexed by  $\{x_1, \dots, x_s\}$  deleted [10]. This will not be feasible for large networks. We therefore resort to a sampling algorithm which computes the collection  $\Gamma$  of cycles and counts  $N_T^\gamma$  from a realization  $(X_n)_{1 \leq n \leq T}$  of the Markov chain  $(X_n)_n$ , see [10]. The algorithm uses an auxiliary chain  $\eta$  which is initialized at  $\eta = X_1$ , then the states  $X_i$  are added sequentially to  $\eta$  until  $\eta$  contains a cycle, which is then removed from  $\eta$ :

**Algorithm summary of the stochastic cycle decomposition:**

- (i) Initialization: Set  $\Gamma = \emptyset$  and  $\eta = X_1$ .
- (ii) for  $i = 2$  to  $T$ : If  $X_i \notin \eta$ , set  $\eta = [\eta, X_i]$ . If  $\eta = [\eta_1, \dots, \eta_{p-1}, X_i, \eta_{p+1}, \dots, \eta_l]$  then
  1. Set  $\gamma = [X_i, \eta_{p+1}, \dots, \eta_l]$  and  $\eta = [\eta_1, \dots, \eta_{p-1}]$ .
  2. If  $\gamma \in \Gamma$ , then  $N_T^\gamma = N_T^\gamma + 1$ . Otherwise add  $\gamma$  to  $\Gamma$  and set  $N_T^\gamma = 1$ .

We then set  $w_T(\gamma) := \frac{1}{T} N_T^\gamma$ . In [10] it has been shown that  $w_T(\gamma) \rightarrow w(\gamma)$  a.s. as  $T \rightarrow \infty$ , therefore the finite-time weights  $w_T$  returned by the sampling algorithm are an approximation of the true weights  $w$ . The computational complexity of this algorithm is at most quadratic in  $T$ . However, the number of cycles with significant weights  $w(\gamma)$  which we have to sample accurately might scale unfavorably with the size of the network. In practise, one has to choose  $T$  such that convergence is observed. If the network is very large and very modular such that  $(X_n)_n$  is very metastable, this might be prohibitively expensive.

**Remark 3** (Timeseries perspective). In many applications, the network  $G$  will be a representation of a finite timeseries of data  $(X_n)_{1 \leq n \leq T}$ . In such situations it

is natural to use the sampling algorithm directly on the data, which fixes  $T$  and removes any convergence issues. Our method then turns into a form of timeseries analysis via recurrence networks. We will address this point of view in more detail in further publications.

**Remark 4** (Alternative Algorithms). Various alternative algorithms exist to obtain a cycle decomposition  $\{\tilde{\Gamma}, \tilde{w}\}$  which satisfies only (3), but not (4). We quickly review two examples:

- (a) **Deterministic iterative algorithm:** [11, 10] This algorithm iterates the following steps:
  - (i) find a cycle  $\gamma$  in  $G$ ,
  - (ii) set  $w(\gamma) = \min_{(xy) \in \gamma} F_{xy}$ ,
  - (iii) update  $F \rightarrow F - w(\gamma)\chi_\gamma$  where  $\chi_\gamma$  is the unit flow along  $\gamma$ .
 The iteration stops when  $F < TOL$  is reached for a prescribed small  $TOL > 0$ . This algorithm has polynomial complexity in the number of vertices of  $G$ .
- (b) **Cycle-basis construction:** [11] The space  $\mathcal{C}$  of cycle flows is a vector space of dimension<sup>2</sup>  $d \leq |E| - |V| + 1$  and can be equipped with a scalar product. Then we may take  $\tilde{\Gamma}$  to be a basis of  $\mathcal{C}$ , which can be found e.g. via a minimal spanning tree [11]. The weights  $\tilde{w}$  can be computed by solving a linear system involving the  $d \times d$  matrix  $C_{\gamma\gamma'} = \langle \chi_\gamma, \chi_{\gamma'} \rangle$ .

If we use (a) or (b) instead of the sampling algorithm in step (2), then the resulting weights  $\tilde{I}_{xy}$  lose their interpretation in terms of information transport measure between  $x$  and  $y$ . Still, in many cases  $\tilde{I}_{xy}$  might be a good approximation to  $I_{xy}$ , and (a) and (b) are fast even for large networks. The approximation quality of  $\tilde{I}_{xy}$  obtained by (a) and (b) will be addressed in our future work.

**3.3. Verification of modularity.** We can use the well-known modularity function [20] to evaluate the resulting partition  $\{\chi_m\}_{m=1}^M$  of  $K_G$ . We obtain that

$$\begin{aligned} \bar{Q} &= \sum_m \sum_{x,y} \chi_m(x) [\mathbf{P}(Z_n = x, Z_{n+1} = y) - \mathbf{P}(Z_n = x)\mathbf{P}(Z_n = y)] \chi_m(y) \\ &= \sum_m \sum_{x,y} \chi_m(x) [I_{xy} - \pi_x \pi_y] \chi_m(y). \end{aligned} \quad (16)$$

Notice that  $\bar{Q}$  is the modularity of the partitioning evaluated on the *undirected* graph  $K_G$ . Recently the extensions of modularity to directed networks have been discussed in literature [15, 12]. These approaches lead to the following modularity function  $Q$  of partitioning on  $G$

$$Q = \sum_m \sum_{x,y} \chi_m(x) [\pi_x p_{xy} - \pi_x \pi_y] \chi_m(y). \quad (17)$$

However, approaches based on such  $Q$  are shown to often neglect important directional information [13, 24] and have the same outcome as when symmetrizing  $P$ . More precisely,  $Q$  is unchanged if  $\pi_x p_{xy}$  is replaced by the symmetrized version  $\pi_x p_{xy}^s = \frac{1}{2}(\pi_x p_{xy} + \pi_y p_{yx})$ , which ignores directions. In contrast, using  $\bar{Q}$  to evaluate the quality of a partitioning of  $K_G$  is unambiguous, and the directional information of  $G$  is directly built into  $I_{xy}$ . We will demonstrate on some examples in Section 4 that  $Q$  and  $\bar{Q}$  may differ significantly. This difference comes mainly from the fact that  $Q$  takes into account only one-step transitions  $p_{xy}$ , unlike  $\bar{Q}$  which considers multi-step transitions encoded in  $I_{xy}$  via cycle structures of  $G$ .

<sup>2</sup>The upper bound on  $d$  is the Betti number of  $G$ .

4. **Numerical experiments.** In this section, we will compare the results of three different clustering algorithms on a few example networks. The algorithms considered are:

1. The algorithm proposed in Section 3, i.e. construction of  $K_G$  via a realization  $(X_n)_{1 \leq n \leq T}$  and then MSM clustering of  $K_G$ , which we will refer to as **Cycle MSM (CMSM) clustering**. This algorithm returns a fuzzy partition  $\{\chi_m\}_{m=1}^M$ .
2. Construct  $K_G$  via a realization  $(X_n)_{1 \leq n \leq T}$  and then optimize the  $K_G$ -based modularity function  $\bar{Q}$  (16) over all full partitions  $\{\chi_m\}_{m=1}^M$ ,  $\chi_m(x) \in \{0, 1\}$ . We will refer to this as  **$\bar{Q}$ -maximization**.
3. Maximize the  $G$ -based modularity function  $Q$  (17) over all full partitions  $\{\chi_m\}_{m=1}^M$ ,  $\chi_m(x) \in \{0, 1\}$ . This direct approach doesn't need the construction of  $K_G$ . We will refer to it as  **$Q$ -maximization**.

4.1. **Barbell graph.** As discussed earlier, the communication graph  $K_G$  consists of two complete graphs with  $n$  nodes each which are joined by the edge  $(l_0 r_0)$ . CMSM clustering produces a full partition  $\{\chi_1, \chi_2\}$  where  $\chi_1$  is one on  $C_1 = \alpha_l$  and zero on  $C_2 = \alpha_r$ , see Figure 6. The scores assigned by  $Q$  and  $\bar{Q}$  to  $\{\chi_1, \chi_2\}$  are almost identical:

$$Q(C_1, C_2) = \frac{1}{2} - \frac{\varepsilon}{n + \varepsilon}, \quad \bar{Q}(C_1, C_2) = \frac{1}{2} - \frac{1}{2} \frac{\varepsilon}{n + \varepsilon}.$$

Now we address the question whether  $\{\chi_1, \chi_2\}$  optimizes  $Q$  resp.  $\bar{Q}$ . To do so, we compare  $\{\chi_1, \chi_2\}$  for even  $n$  with another partition into 3 sets  $C_{1,a}, C_{1,b}$  and  $C_2$  where  $|C_{1,a}| = |C_{1,b}|$ , see Figure 6. Let  $\Delta Q = Q(C_1, C_2) - Q(C_{1,a}, C_{1,b}, C_2)$  and  $\Delta \bar{Q}$  similarly. One finds

$$\Delta Q = \frac{1}{8} - \frac{1}{n + \varepsilon}, \quad \Delta \bar{Q} = \frac{1}{8} - \frac{1}{4} \frac{n}{n + \varepsilon}.$$

Then,  $\Delta Q > 0$  for  $n \geq 8$  and  $\Delta \bar{Q} < 0$  as long as  $n > \varepsilon$ . That is, as  $n$  grows  $Q$ -maximization produces a partition into more and more subchains with less than 8 nodes while  $\bar{Q}$ -maximization always produces the partition  $\{\chi_1, \chi_2\}$ . Notice that the block structure (14) indicates that it is equally easy to transmit information between all nodes in  $C_1$  while it is hard to transmit information between  $C_1$  and  $C_2$ . From the point of view of the RW process this is natural: The average time to go from  $C_1$  to  $C_2$  is  $(n + \frac{1}{2}) \frac{\varepsilon}{1 + \varepsilon}$ , while mixing within  $C_1$  and in particular transitions between  $C_{1,a}$  and  $C_{1,b}$  happen quickly.

To summarize, the barbell graph has a modular structure which is not based on link density and therefore cannot be detected by density based methods such as  $Q$ -maximization. Rather, this modular structure is information based and revealed by  $K_G$ . In the particular case here, using CMSM clustering or  $\bar{Q}$ -optimization to partition  $K_G$  yields the same result.

4.2. **Wheel switch.** Our next example is the 'wheel switch' graph which is shown in Figure 7a. It consists of a clockwise outer loop  $\alpha_o$  and an anticlockwise inner loop  $\alpha_i$  with  $n$  nodes each and some connections between the two loops. At each connection, the probability to switch between  $\alpha_o$  and  $\alpha_i$  is  $p$  and the connections are such that two smaller loops  $\beta_1$  and  $\beta_2$  of length 4 are formed (coloured in Figure 7a).

The community structure of this graph depends on  $p$ . If  $p > 1/2$ , completing the small loops  $\beta_1$  and  $\beta_2$  is more likely than completing  $\alpha_i$  and  $\alpha_o$ , so we expect

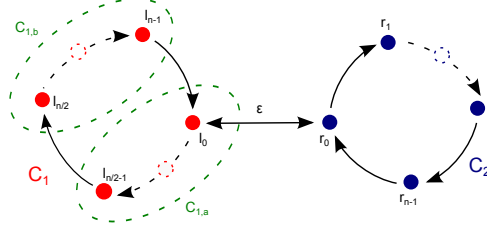


FIGURE 6. The two modules  $C_1$  and  $C_2$  produced by CMSM clustering.  $Q$ -maximization gives a partitioning into subchains of the type  $C_{1,a}, C_{1,b}$ .

a community structure where  $\beta_1$  and  $\beta_2$  dominate. Indeed, CMSM clustering produces the partition shown in Figure 7a which identifies  $\beta_1$  and  $\beta_2$  as modules and gives all other nodes an affiliation of 0.5 to both modules (indicated by the white colour). The modularity values achieved by this partitioning depend on  $p$  and  $n$ , but typical values are  $\bar{Q} \approx 0.2$ .

If  $p \leq 1/2$ , transitions between  $\alpha_i$  and  $\alpha_o$  are unlikely and we expect to find the community structure shown in Figure 7b. Indeed CMSM clustering produces exactly this partition. As in the previous example, for  $p \leq 1/2$   $\bar{Q}$ -maximization gives the same result. On the other hand and again as in the previous example,  $Q$ -maximization divides the red and green modules in Figure 7b further into many small chains of length less than 8.

Also for  $1/2 < p \leq 0.63$ ,  $\bar{Q}$ -maximization produces the partition shown in Figure 7b, while for  $p > 0.63$  it gives the one shown in Figure 7c. Note that the difference between Figure 7a and Figure 7c is only that the white nodes are now fully assigned to one of the two modules in such a way that as much of  $\alpha_i$  and  $\alpha_o$  is intact as possible.  $Q$ -maximization again produces many small chains of length less than 8 as modules, how many depends on  $n$ . Even if we restrict  $Q$ -maximization to partitions with only two modules, the optimum is degenerate: Any cut  $S$  that produces modules of equal size and leaves  $\beta_1$  and  $\beta_2$  intact, as shown in Figure 7d, will optimize  $Q$ .

In summary, CMSM clustering exactly reproduces the expected community structure for all values of  $p$  and  $n$ .  $Q$ -maximization destroys the community structure for large  $n$  and produces many small chains instead.  $\bar{Q}$ -maximization cannot produce the fuzzy partition of Figure 7a, but instead produces the best possible full partition, independent of  $n$ . However, the  $p$ -threshold between outcomes 7a and 7b is  $p = 0.5$  for CMSM clustering, while it is  $p = 0.63$  between outcomes 7b and 7c for  $\bar{Q}$ -maximization.

**5. Conclusion.** In this paper we discussed the problem of finding fuzzy partitions of weighted, directed networks. The main difficulty compared to the well studied case of analyzing undirected networks is that considering asymmetric edge directions often leads to more complicated network substructures. For this reason most of the clustering approaches for undirected networks cannot be generalized to directed networks in a straightforward way. Although different new methods have been proposed, most of them fail to capture all directional information important for module finding, as they consider only one-step, one-directional transitions, which can lead to detecting false modules (where nodes are closely connected only in one



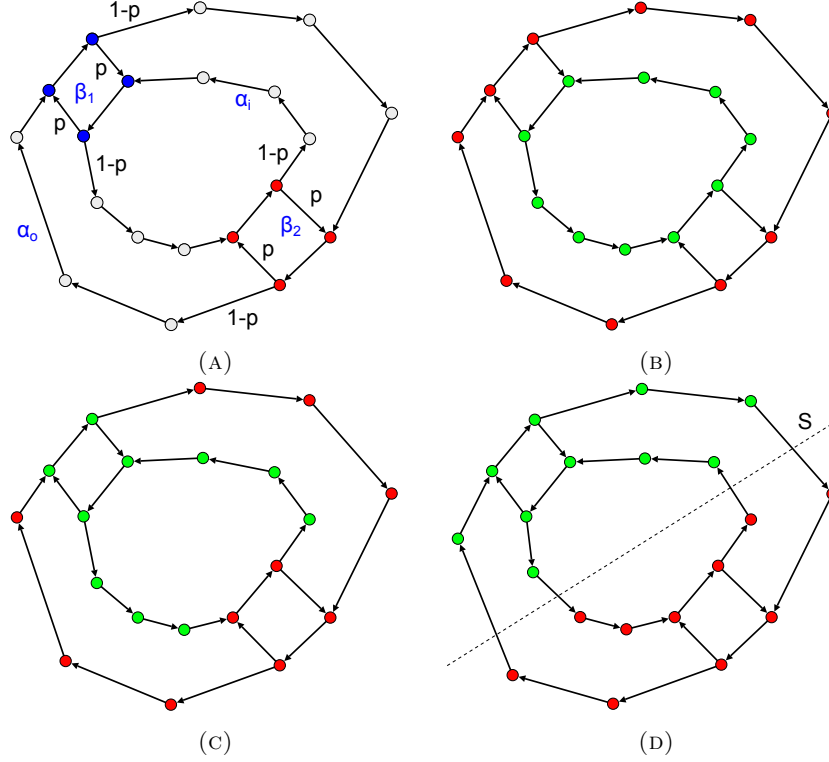


FIGURE 7. Graph of the wheel switch with  $n = 10$ , and  $p$  being the probability to switch between the outer and inner loop. The coloring: in (a) shows the modules found by CMSM clustering for  $p > 1/2$ ; in (b) is shown clustering found by CMSM for  $p \leq 1/2$ ; (c) shows resulting clustering of  $\bar{Q}$ -maximization for  $p \geq 0.63$ ; and (d) one example of clustering obtained by  $Q$ -maximization for  $p \geq 1/2$ .

direction) and over-partitioning of the network. To this end, we introduced herein a novel measure of communication between nodes which is based on using multi-step, bidirectional transitions encoded by a cycle decomposition of the probability flow. We have shown that this measure is symmetric and that it reflects the information transport in the network. Furthermore, this enabled us to construct an undirected communication graph that captures information flow of the original graph and apply clustering methods designed for undirected graphs. We applied our new method to several examples, showing how it overcomes the essential limitation of common methods. This indicates that our method can be used to circumvent the problems of one-step methods, like modularity based methods. Further generalizations of modularity function and consideration of different null models will be the topic of our future research. An important aspect of our method is its role in describing non-equilibrium Markov processes and in particular the connection with entropy production rate. Our future research will be oriented towards these problems.

**Acknowledgments.** The authors would like to thank Peter Koltai and Marco Sarich for insightful discussions. RB thanks the Dahlem Research School (DRS) and the Berlin Mathematical School (BMS).

## Appendix.

**Proof of equation 7:** We want to find the transition matrix for  $(\xi_n)_n$ :

$$\begin{aligned} \mathbf{P}(\xi_n = \beta | \xi_{n-1} = \alpha) &= \sum_x \mathbf{P}(\xi_n = \beta | \xi_{n-1} = \alpha, X_n = x) \mathbf{P}(X_n = x | \xi_{n-1} = \alpha) \\ &= \sum_x \mathbf{P}(\xi_n = \beta | X_n = x) \mathbf{P}(X_n = x | \xi_{n-1} = \alpha) \\ &= \sum_{x \in (\alpha \cap \beta)} b_\beta^{(x)} \mathbf{P}(X_n = x | \xi_{n-1} = \alpha) \end{aligned}$$

by properties of  $(X_n)_n$ ,  $(\xi_n)_n$  and the definition of  $b_\beta^{(x)}$ . Now

$$\begin{aligned} \mathbf{P}(X_n = x | \xi_{n-1} = \alpha) &= \sum_y \mathbf{P}(X_n = x | \xi_{n-1} = \alpha, X_{n-1} = y) \mathbf{P}(X_{n-1} = y | \xi_{n-1} = \alpha) \\ &= \sum_y \mathbf{P}(X_n = x | \xi_{n-1} = \alpha, X_{n-1} = y) \\ &\quad \times \mathbf{P}(\xi_{n-1} = \alpha | X_{n-1} = y) \frac{\mathbf{P}(X_{n-1} = y)}{\mathbf{P}(\xi_{n-1} = \alpha)} \\ &= \sum_{y \subset \alpha} \mathbf{P}(X_n = x | \xi_{n-1} = \alpha, X_{n-1} = y) b_\alpha^{(y)} \frac{\pi_y}{|\alpha| F(\alpha)} \\ &= \sum_{y \subset \alpha} \mathbf{P}(X_n = x | \xi_{n-1} = \alpha, X_{n-1} = y) \frac{1}{|\alpha|} \end{aligned}$$

Since  $(X_n)_n$  is actually determined by the pair  $(\xi_{n-1}, X_{n-1})$ , we have

$$\mathbf{P}(X_n = x | \xi_{n-1} = \alpha, X_{n-1} = y) = \begin{cases} 1 & y = n_\alpha^-(x) \\ 0 & \text{otherwise} \end{cases}$$

where  $n_\alpha^-(x)$  is the vertex we reach next if we follow the cycle  $\alpha$  in reverse orientation starting at  $x$ . Putting it all together, we have

$$\mathcal{Q}_{\alpha\beta} := \mathbf{P}(\xi_n = \beta | \xi_{n-1} = \alpha) = \sum_{x \in (\alpha \cap \beta)} \frac{1}{|\alpha|} b_\beta^{(x)}.$$

**Proof of Lemma 2.5:** (i) follows from (ii) by the fact that  $\mathcal{P}$  and  $\mathcal{Q}$  are stochastic. To show (ii),  $\pi_x \mathcal{P}_{xy} = \pi_y \mathcal{P}_{yx}$  and  $\mu(\alpha) \mathcal{Q}_{\alpha\beta} = \mu(\beta) \mathcal{Q}_{\beta\alpha}$  can be directly inferred from (11) and (12). (iii) follows from (iv). To show (iv), let  $\lambda \neq 0$  and  $v \in \ker(\mathcal{Q} - \lambda 1)$ . This implies  $\mathcal{V}\mathcal{B}v = \lambda v$ , multiplication from the left with  $\mathcal{B}$  yields  $\mathcal{P}\mathcal{B}v = \lambda \mathcal{B}v$ , hence  $\mathcal{B}v \in \ker(\mathcal{P} - \lambda 1)$ . Now since  $\mathcal{Q} = \mathcal{V}\mathcal{B}$  we have  $\ker \mathcal{B} \subset \ker \mathcal{Q} \perp \ker(\mathcal{Q} - \lambda 1)$ . Therefore  $\mathcal{B}$  is injective as a map from  $\ker(\mathcal{Q} - \lambda 1)$  to  $\ker(\mathcal{P} - \lambda 1)$ . To show that  $\mathcal{B}$  is also surjective, take  $w \in \ker(\mathcal{P} - \lambda 1)$ . Define  $v := \frac{1}{\lambda} \mathcal{V}w$ . Then  $v \in \ker(\mathcal{Q} - \lambda 1)$  since  $\mathcal{Q}v = \frac{1}{\lambda} \mathcal{Q}\mathcal{V}w = \frac{1}{\lambda} \mathcal{V}\mathcal{P}w = \frac{1}{\lambda} \mathcal{V}\lambda w = \lambda v$ , and  $\mathcal{B}v = \frac{1}{\lambda} \mathcal{B}\mathcal{V}w = w$ .

## REFERENCES

- [1] A Fernández A Arenas<sup>1</sup>, J Duch and S Gómez. Size reduction of complex networks preserving modularity. *New J. Phys.*, 9:176, 2007.
- [2] B. Altaner, S. Grosskinsky, S. Herminghaus, L. Katthän, M. Timme, and J. Vollmer. Network representations of nonequilibrium steady states: Cycle decompositions, symmetries, and dominant paths. *Phys. Rev. E*, 85:041133, 2012.
- [3] A. Barrat, M. Barthelemy, R. Pastor-Satorras, and A. Vespignani. The architecture of complex weighted networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(11):3747–3752, 2004.
- [4] Jingchun Chen and Bo Yuan. Detecting functional modules in the yeast protein–protein interaction network. *Bioinformatics*, 22(18):2283–2290, September 2006.
- [5] N. Djurdjevac, S. Bruckner, T. O. F. Conrad, and Ch. Schütte. Random walks on complex modular networks. *Journal of Numerical Analysis, Industrial and Applied Mathematics*, 6:29–50, 2011.
- [6] N. Djurdjevac, M. Sarich, and Ch Schütte. Estimating the eigenvalue error of markov state models. *Multiscale Modeling & Simulation*, 10:61–81, 2012.
- [7] T. S. Evans and R. Lambiotte. Line graphs, link partitions, and overlapping communities. *Phys. Rev. E*, 80:016105, 2009.
- [8] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3–5):75 – 174, 2010.
- [9] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–7826, 2002. cited By (since 1996)3239.
- [10] D. Jiang, M. Qian, and M.-P. Quian. *Mathematical theory of nonequilibrium steady states: on the frontier of probability and dynamical systems*. Springer, 2004.
- [11] Sophia L. Kalpazidou. *Cycle Representations of Markov Processes*. Springer, 2006.
- [12] Youngdo Kim, Seung-Woo Son, and Hawoong Jeong. Finding communities in directed networks. *Phys. Rev. E*, 81:016103, Jan 2010.
- [13] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E*, 80:016118, Jul 2009.
- [14] Andrea Lancichinetti, Filippo Radicchi, José J. Ramasco, and Santo Fortunato. Finding statistically significant communities in networks. *PLoS ONE*, 6(4):e18961, 04 2011.
- [15] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Phys. Rev. Lett.*, 100:118703, Mar 2008.
- [16] Fragkiskos D. Malliaros and Michalis Vazirgiannis. Clustering and community detection in directed networks: A survey. *Physics Reports*, 533(4):95 – 142, 2013. Clustering and Community Detection in Directed Networks: A Survey.
- [17] P. Metzner, Ch. Schütte, and E. Vanden-Eijnden. Transition path theory for markov jump processes. *Multiscale Modeling & Simulation*, 7(3):1192–1219, 2009.
- [18] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.
- [19] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, 2004.
- [20] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.
- [21] Prot Pakoński, Gregor Tanner, and Karol Życzkowski. Families of line-graphs and their quantization. *Journal of Statistical Physics*, 111(5-6):1331–1352, 2003.
- [22] Dragoš Cvetkovic, Peter Rowlinson, and Slobodan Simic. *Spectral Generalizations of Line Graphs*. Cambridge University Press, 2004. Cambridge Books Online.
- [23] M. Sarich, N. Djurdjevac, S. Bruckner, T. O. F. Conrad, and Ch. Schütte. Modularity revisited: A novel dynamics-based concept for decomposing complex networks. *Journal of Computational Dynamics*, 1(1):191–212, 2014.
- [24] Michael T. Schaub, Jean-Charles Delvenne, Sophia N. Yaliraki, and Mauricio Barahona. Markov dynamics as a zooming lens for multiscale community detection: Non clique-like communities and the field-of-view limit. *PLoS ONE*, 7(2):e32210, 02 2012.
- [25] Michael T. Schaub, Renaud Lambiotte, and Mauricio Barahona. Encoding dynamics for multiscale community detection: Markov time sweeping for the map equation. *Phys. Rev. E*, 86:026112, Aug 2012.

- [26] J. Schnakenberg. Network theory of microscopic and macroscopic behavior of master equation systems. *Rev. Mod. Phys.*, 48:571–585, Oct 1976.
- [27] Alcides Viamontes Esquivel and Martin Rosvall. Compression of flow can reveal overlapping-module organization in networks. *Phys. Rev. X*, 1:021025, Dec 2011.
- [28] R. K. P. Zia and B. Schmittmann. Probability currents as principal characteristics in the statistical mechanics of non-equilibrium steady states. *Journal of Statistical Mechanics-theory and Experiment*, 7, 2007.

*E-mail address:* [djurdjev@mi.fu-berlin.de](mailto:djurdjev@mi.fu-berlin.de)

*E-mail address:* [ralf.banisch@fu-berlin.de](mailto:ralf.banisch@fu-berlin.de)

*E-mail address:* [Christof.Schuette@fu-berlin.de](mailto:Christof.Schuette@fu-berlin.de)